PCT/GB 9 8 / 0 3 7 5 3

25 FEBRUARY 1999.

Europäisches Patentamt | European Patent Office | Office européen des brevets

Bescheinigung | Certificate | Attestation

Die angehefteten Unterlagen stimmen mit der ursprünglich eingereichten Fassung der auf dem nächsten Blatt bezeichneten europäischen Patentanmeldung überein.

The attached documents are exact copies of the European patent application described on the following page, as originally filed.

Les documents fixés à cette attestation sont conformes à la version initialement déposée de la demande de brevet européen spécifiée à la page suivante.

Patentanmeldung Nr. | Patent application No. | Demande de brevet n°

97310358.3

Der Präsident des Europäischen Patentamts:
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

H.I. Block

DEN HAAG, DEN
THE HAGUE,
LA HAYE

22/01/99

Europäisches Patentamt
European Patent Office
Office européen des brevets

Blatt 2 der Bescheinigung
Sheet 2 of the certificate
Page 2 de l'attestation

Anmeldung Nr.:
Application no.: 97310358.3
Demande n°:

Anmeldetag:
Date of filing: 19/12/97
Date de dépôt:

Anmelder:
Applicant(s):
Demandeur(s):
BRITISH TELECOMMUNICATIONS public limited company
London EC1A 7AJ
UNITED KINGDOM

Bezeichnung der Erfindung:
Title of the invention:
Titre de l'invention:
Data communications

In Anspruch genommene Priorität(en) / Priority(ies) claimed / Priorité(s) revendiquée(s)

Staat: State: Pays:	Tag: Date: Date:	Aktenzeichen: File no. Numéro de dépôt:

Internationale Patentklassifikation:
International Patent classification:
Classification internationale des brevets:
H04L12/14, H04L29/06

Am Anmeldetag benannte Vertragstaaten:
Contracting states designated at date of filing: AT/BE/CH/[redacted]/DE/DK/ES/FI/FR/GB/GR/IE/IT/LI/LU/MC/NL/PT/SE
Etats contractants désignés lors du depôt:

Bemerkungen:
Remarks:
Remarques:

DATA COMMUNICATIONS

The present invention relates to a data communications system, and in particular to a system using an extensive public network such as the Internet.

Hitherto, users have typically been charged for Internet usage following a subscription model in which they pay a flat-rate fee to an Internet Services Provider (ISP) for access to the Internet and use of network resources is subsequently free of charge. The different networks and routers making up the Internet have then endeavoured to meet the need of any particular application on a "best effort" basis. It has been recognised that this method of operation is not appropriate for applications such as multimedia conferencing which use a lot of network resources, and which ideally require a guaranteed quality of service (QoS). A customer for such services may therefore pay a premium to an Internet service provider in order to receive a guaranteed QoS for a given session, or over an extended period of time. More generally, there may be a need to charge customers according to the amount of data transmitted. This might be used, for example, to provide a charging mechanism for access to on-line data sources, such as a video library.

Where the customer is paying for a certain specified quality of service, or for a certain amount of data received, there arises the problem of resolving any disputes between the customer and the service provider as to the level of service the customer has received in a given period. A customer might, for example, pay for an enhanced QoS level for a session of video conferencing. If however the network then fails to provide the required QoS, for example because a high proportion of data packets have been lost, then the customer might claim a refund. However, while conventional networks, such as the PSTN, incorporate extensive and reliable billing systems which carefully record details of all calls and generate reliable records, no such billing/auditing structure exists within the Internet. Moreover, it would be undesirable to incorporate a conventional billing structure in the Internet or any other similar public data network, since this would add considerably to the operational costs of the network. There remains a need therefore for alternative mechanisms for resolving any disputes between parties as to the quality of service which has been delivered.

According to a first aspect of the present invention, there is provided a method of operating a data communications system comprising:

a) at a remote data source, outputting a plurality of data frames

b) encrypting the data frames;

c) transmitting the data frames via a communications network to a customer terminal;

d) at the customer terminal, decrypting the data frames;

e) storing a record of the data frames decrypted in step (d); and

f) subsequently generating a receipt for data frames received at the customer terminal by reading record data stored in step (e) .

The present invention provides a new approach to the generation of an auditable record of the data received by a customer connected to a data communications network. The data source transmits the data as a series of data frames. These frames are typically application-level entities, and need not correspond to the frame structure, if any, of the transmission layer of the network. The frames might correspond, for example, to successive minutes of video delivered from a video server or to individual strokes on a white board in a conferencing application. Each successive data frame is then encrypted. Therefore, in order for the customer to be able to use the data, each frame has to be decrypted. A record is kept of the decryption of the data frames. This record may comprise a count of the number of frames decrypted. This record may itself be encrypted. This then provides a verifiable record which can be used, for example, to resolve disputes as to the number of data frames received in a given time period. As in the detailed embodiment below, different keys may be used for encryption and decryption of different frames. The record of the decryption of the frames may then be derived by storing a count of the number of keys generated at the customer terminal. Other data may also be stored, such as the time of the session and/or the time at which each key is generated.

The customer terminal may be a personal computer or any other appropriate device, such as, for example, a Java-enabled mobile cellular telephone.

Preferably the steps of storing a record (and optionally of generating a plurality of different keys) are carried out by a secure module located at the customer terminal. The secure module provides a region in the customer terminal which is effectively under the control of the data provider, and which is not readily

accessible to the customer. The use of such a secure module further enhances the reliability of the stored record. The secure module may be a software module which executes a cryptographic algorithm. This might be implemented, for example, as a Java program distributed by the operator of the remote data source as part of the process of setting up a session. To provide still higher levels of security, it is preferred that the secure module should include a dedicated processor and store located within a tamper-proof unit. Examples of such secure modules include smartcard structures, and cryptographic PC cards.

When the secure module has only a relatively low processing power, as may be the case, for example, when it is a smartcard, then preferably that module is required simply to output the different respective keys. Other processes running in the main part of the customer terminal are then responsible for decrypting the data frames. Alternatively, when the secure module has more processing power, as when, for example, a cryptographic co-processor card is used, then preferably the encrypted data frames are passed to the secure module and the module generates a key, decrypts the frames, and passes the decrypted frames out, for example, to an application programme running on the customer terminal. In this case it is not necessary to generate a new key for each frame since the key is kept within the secure module.

Preferably the remote data source generates and transmits to the customer terminal a seed value, and the plurality of different keys are generated from the said seed value. A fresh seed value may be used for each session.

The data frames from the remote data source may be multicast to a plurality of different customer terminals. In this case preferably seed values for the generation of the plurality of different keys are distributed to the plurality of terminals.

Preferably a digital watermark is added to the data frames in the said step of decrypting the data frames. Digital watermarking is a well-known technique whereby, for example, insignificant bits of a digital data stream may be varied in a characteristic fashion. If the data is then copied and passed on, the source can be identified by inspecting the insignificant bits. The use of digital watermarking is particularly valuable in the context of the present invention, since it facilitates detection of attempts to undermine the security of the system by collusion between two or more customers, for example by one customer decrypting and

retransmitting frames.

According to a second aspect of the present invention, there is provided a data communication system comprising:

a) a remote data source arranged to output a plurality of frames;

b) encryption means for encrypting a plurality of frames with different respective keys;

c) a communications network connected to the encryption means;

d) a customer terminal connected to the communications network and arranged to receive encrypted data frames via the communications network;

e) decryption means located at the customer terminal and arranged to generate a plurality of different keys for decrypting different respective data frames; and

f) a store at the customer terminal for storing a record of keys generated by the decryption means.

According to another aspect of the present invention, there is provided a method of operating a data communications system comprising:

a) at a remote data source, outputting a plurality of data frames;

b) encrypting different ones of the plurality of data frames using different respective keys;

c) transmitting the data frames via a communications network to a customer terminal;

d) at the customer terminal, generating a plurality of different keys for decrypting different respective data frames received at the customer terminal; and

e) storing a record of the keys generated.

Methods and apparatus embodying the present invention will now be described in further detail, by way of example only, with reference to the accompanying drawings in which;

Figure 1 is a schematic of a data communication system embodying the network;

Figure 2 is a schematic showing in further detail the functional components of the customer terminal in the system of Figure 1;

Figure 3 is a flow diagram showing the principal phases of operation of the

system of Figure 1;

Figure 4 is a flow diagram showing in further detail the verification phase;

Figure 5 is a flow diagram showing in further detail the initialisation phase;

Figure 6 is a flow diagram showing in further detail the received/decrypt phase;

Figure 7 is a flow diagram showing in further detail the receipt phase.

A data communications system includes a data server 1 connected via a data communications network 2 to a number of customer terminal 3. Although for ease of illustration only two customer terminals are shown, in practice the data server 1 may communicate simultaneously with many terminals. In the present example, the data communications network 2 is the public Internet and is formed from a number of sub-networks 2a-2d. The sub-networks and the associated routers support IP (Internet Protocol) multicasting.

In the present example, the data server 1 is a video server. The data server reads a video data stream from a mass storage device and compresses the data using an appropriate compression algorithm such as MPEG 2. An encryption module in the data server 1 then divides the compressed video data stream into frames. For example each frame may comprise data corresponding to one minute of the video signal. An encryption algorithm then encrypts the frames of data. Suitable encryption algorithms include DES (Data Encryption Standard) (US Federal Standard FIPS PUB46). This is a conventional private key algorithm. A common encryption algorithm is used for all of the frames in one session. In this embodiment, a sequence of keys is used, with a different key for each successive frame. (The frame referred to in this embodiment is an application-level entity created for the purposes of encryption and is to be distinguished from conventional video "frames").

At each customer terminal, incoming data frames are processed using a secure module 4. As described in further detail below, the secure module 4 generates a sequence of keys corresponding to those used originally to encrypt the data frames. The keys may be passed out to the main processor of the customer terminal to allow the data to be decrypted. Alternatively, the secure module itself may carry out the step of decryption. In either case, the secure module stores a record of the decryption of the data frames. This record may comprise, for example, a count of the total number of keys issued in the course of a session and

hence of the number of frames decrypted, together with a session ID and a record of the time of the session.

Prior to commencing a session, a customer terminal 3 may have contracted with the operator of the data network 2 for a quality of service (QoS) which requires a specified minimum number of frames to be delivered per unit time. If subsequently, congestion in the network 2 causes the rate of frame delivery to fall below that specified in the contract, then the customer terminal 3 request from the data server 1 a refund of charges for the session. To validate this request, the data server 1 requests from the secure module 4 a "receipt". This receipt includes the data recorded in the data store and so provides a tamper-proof indication of the number of frames decrypted and made available to the customer in the course of a specified session.

Figure 2 shows the principal functional components of the customer terminal relevant to the present invention. A network interface 22 communicates data frames to and from the data network. The data frames pass from the interface 22 to a secure module 23. The secure module 23 has sub-modules comprising a decryption module D a key generation module K and a secure store S. The key generation module passes a series of keys to the decryption module which decrypts a series of data frames received from the interface 22 and passes these to an application layer module 24. This carries out further processing and passes the resulting data to an output device, which in this example is a video display unit VDU 25. In a preferred implementation, the interface 22 may be embodied in hardware by an ISDN modem and in software by a TCP-IP stack. The secure module 23 may be, for example, a smartcard which is interfaced to the customer terminal via a PCMCIA socket. Suitable smartcards are available commercially from Gemplus and other companies. The smartcard may use one of a number of standard data interfaces such as the Java card API (application programmer's interface) of Sun Microsystems, or the Microsoft smartcard architecture. Alternatively, the secure module may be embodied by a PCI cryptographic co-processor card such as that available commercially from IBM.

Figure 3 shows the main phases in the operation of the system described above. In phase P1, the server verifies that the secure module in the customer terminal is trustworthy and has a recognised identity. In phase P2 the secure

module is initialised to decode data for a particular session. In phase P3 the data is transmitted and decryption carried out and in stage P4, which is optional, a receipt is generated. These phases will now be described in further detail.

When the secure module is, for example, a smartcard, then that smartcard is issued by the manufacturer with a unique public/private key pair. This key pair may be certified by a trusted third party. In phase P1, the server carries out steps to confirm that the smartcard does indeed come from a trusted supplier. The steps of phase P1 are shown in figure 4. In step S1 the server generates a random string. In step S2, the server sends the random string via the data network to the customer terminal. In step S3, the random data string is passed to the secure module (e.g. the smartcard). In step S4 the smartcard signs the random string with its private key. In step S5 the smartcard returns the signed string together with its relevant public key (which has itself been signed by the trusted third party) to the client application running on the customer terminal. In step S6, that client application returns the signed string and the signed public key via the data communications network to the server. In step S7 the server verifies the signed random string.

As shown in Figure 5, to set up the secure module to decode data in a particular session, the server first generates (s51) a seed value for use with an appropriate pseudo-random or chaotic function to generate a series of keys. It also generates a session key (s52). The server encrypts the seed value using the secure module's public key (s3). It then transmits the encrypted seed value and the session key to the customer terminal (s54). The client application passes the seed value and session key on to the secure module (s55). The secure module sets a packet counter to zero (s56) and initialises a sequence generator with the seed value (s57). The customer terminal is then ready to receive and decrypt data frames.

The server subsequently sends a series of frames to the client. Each frame has a frame number (also termed herein the packet number). Each frame might also have a session key transmitted with it. The sequence of steps for the nth frame is illustrated in Figure 6. In step s61 the server sends the encrypted nth frame to the client. The client requests the key x for frame n from the secure module (s62). The secure module records the request (s63). The smartcard then returns the key x to the client (s64). The client deciphers the frame using x (s65).

The client tests to determine whether the frame is the last of a session (s66). If not then the steps are iterated for the n + 1th and subsequent frames.

In setting up the session, the customer has previously negotiated an agreement with the service provider as to the QoS level for the session. For an application such as video on demand this level may be stringent: for example the customer may require that no application-level frame is lost in transmission. If then this QoS level is not met, then the customer requests a refund from the service provider. The request for refund might specify, for example, that there was frame loss at a specified time into the video transmission. In processing such a request, the server requires a receipt from the customer. As shown in Figure 7, in step s71 the client requests a receipt for a specified session s from the secure module. The secure module reads the data which it recorded for that session and generates a receipt containing that data (s72). The secure module signs the receipt with the secure module's private key (s73). The secure module returns the signed receipt to the client (s74). The client in turn transmits the signed receipt to the server (s75). The server checks the signature on the receipt using the public key of the secure module (s76). The public key may be read from a database stored at the server. Having verified the signature, the server can then check the customers claim for a refund using the data contained in the receipt. This data may show, for example, a discrepancy between the number of frames decrypted in a session and the number transmitted by the server, thereby substantiating the customer's claim that a frame was lost.

The sequence used for generating the keys in the above example may be distributed to customers terminals using HTTP (hypertext transfer protocol) as Java code. A suitable chaotic function is:

$$x_{n+1} = 4rx_n(1-x_n)$$

When $r = 1$ this function takes and generates numbers in the range 0 to 1. A chaotic function such as this has the property that any errors in the value of xn grow exponentially as the function is iterated. In use, the secure module uses a higher accuracy internally than the accuracy of the key values exposed to the client. For example the secure module may use 128-bit numbers internally and then only return to the client the most significant 32 bits. In generating the key

values, the chaotic function is iterated until the error in the va client grows bigger than the range. This then prevents the u sequence from the values returned by the secure module.

As an alternative or additional security measure, a different function for each session. This serves to further reduce the possibility of th predicting sequence values.

Table 1 below list Java code for implementing a chaotic function. It returns next number in a sequence, or the nth number in a sequence.

```
/** Class to implement a chaotic sequence */

public class SecureSequence {

  protected int seqNum;
  protected double currNum;


  /** Create a SecureSequence object from a new seed */

  public SecureSequence (double currNum) {
    seqNum = 0;
    this.currNum = currNum;
  }


  /** Return the next number in the sequence */

  public int next() {
    ++seqNum;

    for (int i = 0; i < 20; ++i)  // 20 iterations is a guess,
could use less
```

```
        currNum = 4 * currNum * (1 - currNum);

      // return the most significant 32 bits of a 64 bit number

      return (int)((double)Integer.MAX_VALUE * currNum);

    }


  /** Return the current sequence number of the last number
returned */

    public int sequenceNumber() {
      return seqNum;
    }


    /** Return the number in the sequence at the requested
position in
        the sequence */

    public int next(int seqNum) {

      // if the number is too small return zero (should really be
an exception)

      if (seqNum <= this.seqNum)
        return 0;

      // iterate through the sequence to get to the right number

      while (this.seqNum != seqNum)
        int value = next();

      return value;0
    }
```

}

CLAIMS

1. A method of operating a data communications system comprising:

a) at a remote data source, outputting a plurality of data frames;

b) encrypting the data frames;

c) transmitting the data frames via a communications network to a customer terminal;

d) at the customer terminal, decrypting the data frames;

e) storing a record of the data frames decrypted in step (d); and

f) subsequently generating a receipt for data frames received at the customer terminal by reading record data stored in step (e) .

2. A method according to claim 1, in which step (e) is carried out by a secure module located at the customer terminal.

3. A method according to claim 2, in which the secure module comprises a dedicated processor and a store both located within a tamper-proof unit.

4. A method according to claim 2 or 3 in which the encrypted data frames are passed to the secure module, and the secure module outputs decrypted data frames.

5. A method according to any one of the preceding claims, in which each of a plurality of frames output by the data source is encrypted with a different key, and a plurality of corresponding keys are generated at the customer terminal.

6. A method according to claim 5 when dependent on any one of claims 1 to 3, in which the secure module outputs the plurality of corresponding keys and the customer terminal uses the said keys to decrypt the plurality of frames.

7. A method according to claim 5 or 6, in which the remote data source generates and transmits to the customer terminal a seed value and the plurality of different keys are generated from the said seed value.

8 A method according to any one of the preceding claims, including applying different characteristic variations to data decrypted at different respective customer terminals.

9. A method according to any one of the preceding claims including returning the receipt to the server .

10. A data communications system comprising

a) a remote data source arranged to output a plurality of frames;

b) encryption means for encrypting the plurality of frames;

c) a communications network connected to the encryption means.

d) a customer terminal connected to the communications network and arranged to receive encrypted data frames via the communications network;

e) decryption means located at the customer terminal and arranged to decrypt the data frames received at the customer terminal from the communications network;

f) a store at the customer terminal for storing a record of data frames decrypted by the decryption means;

g) means for reading record data from the store and generating thereby a receipt for data frames received and decrypted by the customer terminal.

11. A data communications system according to claim 10, in which the communications network is a packet-switched network.

12. A data communications system according to claim 10 or 11, in which the store is located in a secure module in the customer terminal.

13. A data communications system according to claim 12, in which the secure module comprises a tamper-proof unit.

14. A data communications system according to claim 12 or 13, in which the encryption means are arranged to encrypt different data frames with different

respective keys, and the secure module is arranged to generate a plurality of keys for decrypting the plurality of frames received at the customer terminal.

15. A customer terminal for use in a method according to any one of claims 1 to 9, the customer terminal comprising:

a) a data interface for connection to a data communications network;

b) decryption means connected to the data interface and arranged to decrypt data frames received via the data interface;

c) a store containing a record of data frames decrypted by the decryption means; and

d) means for reading record data from the store and generating thereby a receipt for data frames received and decrypted by the decryption means.

16. A customer terminal according to claim 15, in which the store is located in a secure module.

17. A customer terminal according to claim 16, in which the secure module comprises a tamper-proof unit.

18. A customer terminal according to claim 16 or 17, in which the secure module is arranged to generate a plurality of keys for decrypting the plurality of frames received at the customer terminal.

19. A method of operating a data communications system comprising:

a) at a remote data source, outputting a plurality of data frames

b) encrypting different ones of the plurality of data frames using different respective keys;

c) transmitting the data frames via a communications network to a customer terminal;

d) at the customer terminal, generating a plurality of different keys for decrypting different respective data frames received at the customer terminal;

e) storing a record of the keys generated.

20. A data communications system comprising

a) a remote data source arranged to output a plurality of frames;

b) encryption means for encrypting the plurality of frames with different respective keys;

c) a communications network connected to the encryption means.

d) a customer terminal connected to the communications network and arranged to receive encrypted data frames via the communications network;

e) a key generator programmed to generate a sequence of keys for use in decrypting data frames:

f) decryption means connected to the key generator and arranged to decrypt the data frames received at the customer terminal from the communications network; and

g) a store at the customer terminal for storing a record of keys generated by the decryption means.

21. A customer terminal for use in a method according to claim 19, the customer terminal comprising:

a) a data interface for connection to a data communications network;

b) a key generator programmed to generate a sequence of keys for use decrypting data frames:

c) decryption means connected to the data interface and to the key generator and arranged to decrypt data frames received via the data interface;

d) a store containing a record of keys generated by the key generator; and

e) means for reading record data from the store and generating thereby a receipt for data frames received and decrypted by the decryption means.

ABSTRACT

Data Communications

In a data communications system a remote data source outputs data as a series of frames. Each frame is individually encrypted with a different key. The keys are transmitted (for example using Internet multicasting) via a communications network to one or more customer terminals. At the terminals a sequence of keys is generated for use in decrypting the frames. A record is kept of the keys generated, and this record may subsequently be used to generate a receipt for the data received by the customer. The keys may be generated, and the record stored, within a secure module such as a smartcard.

Figure (1)

Figure 1

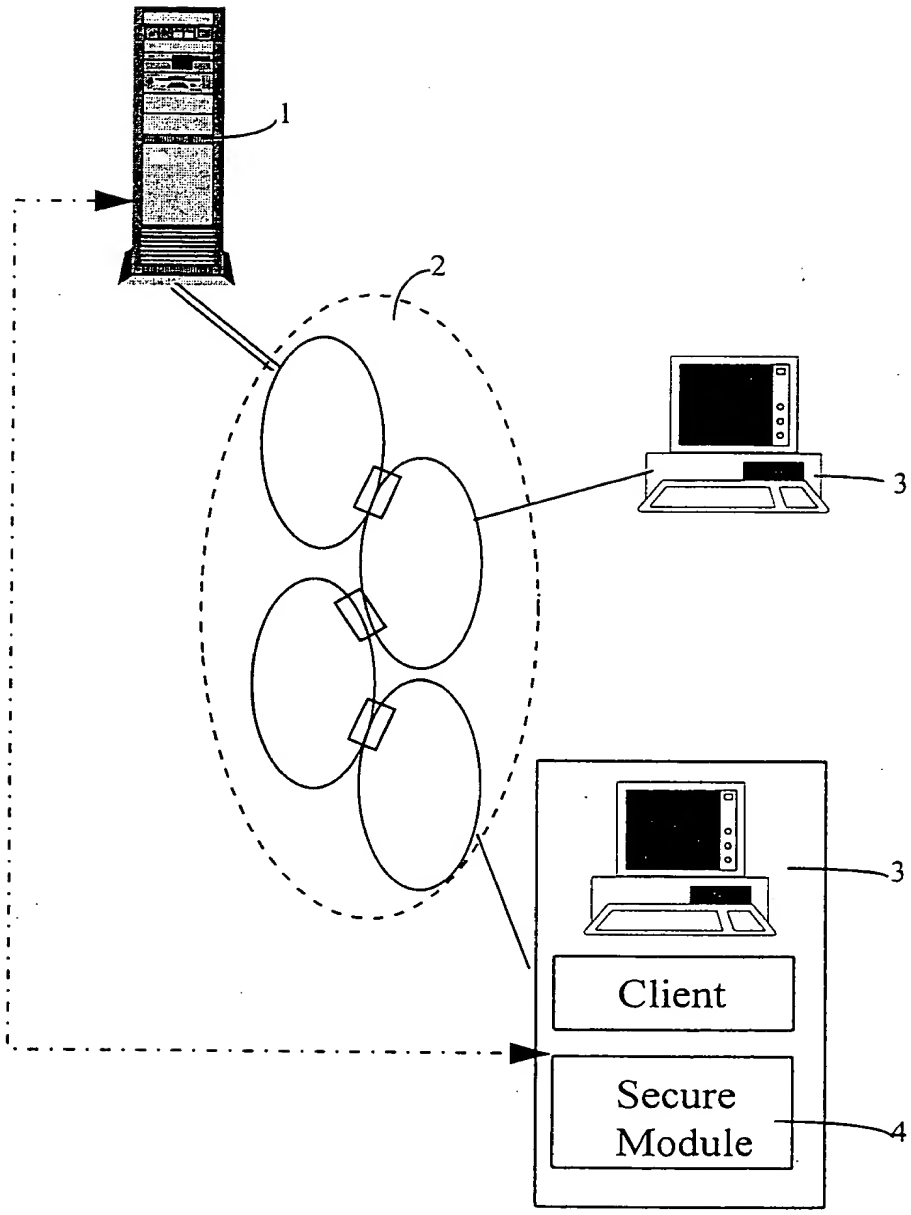


Figure 2

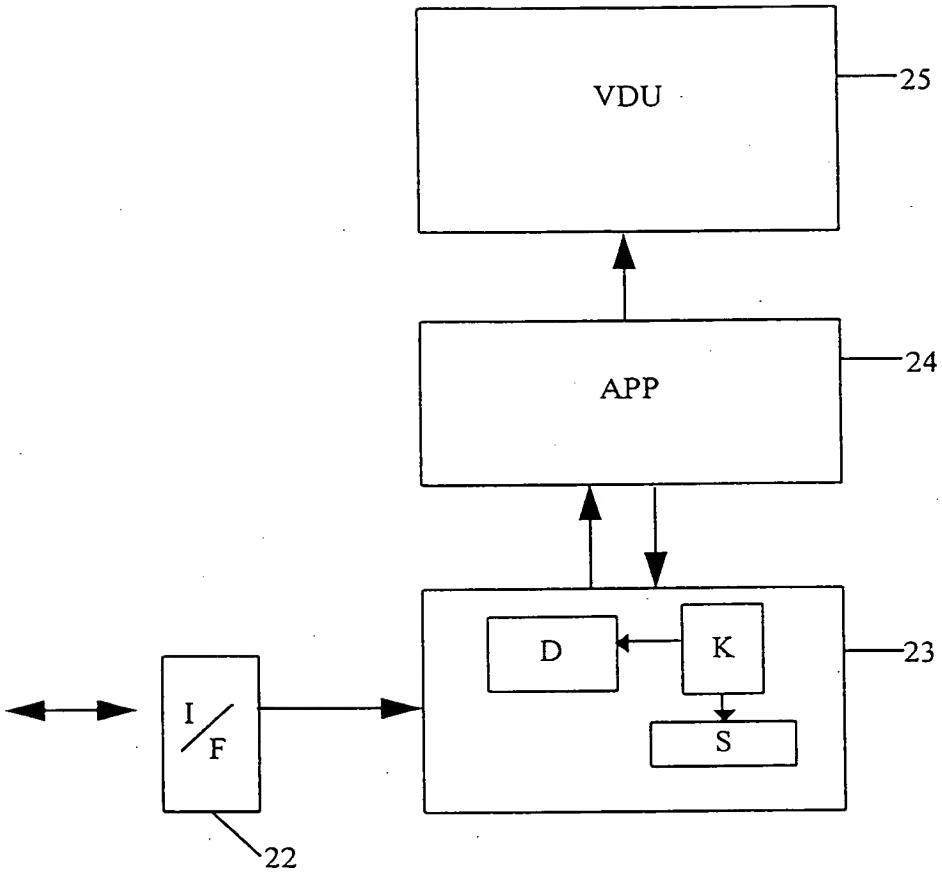


Figure 3

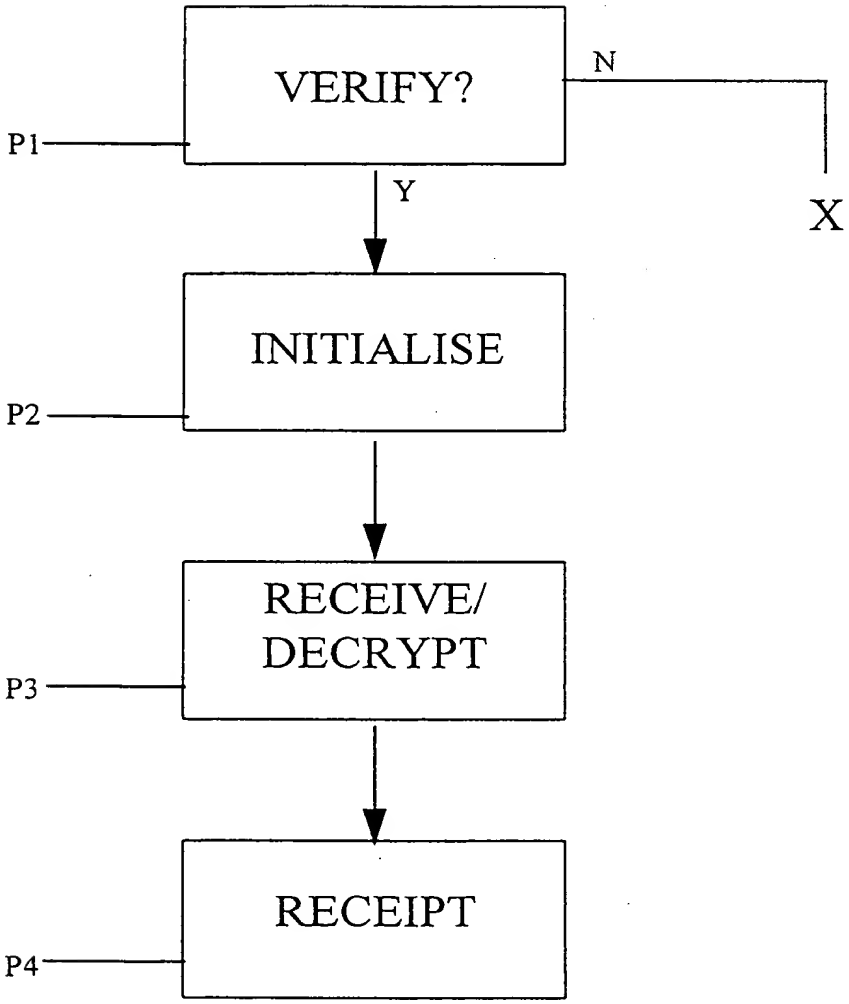


Figure 4

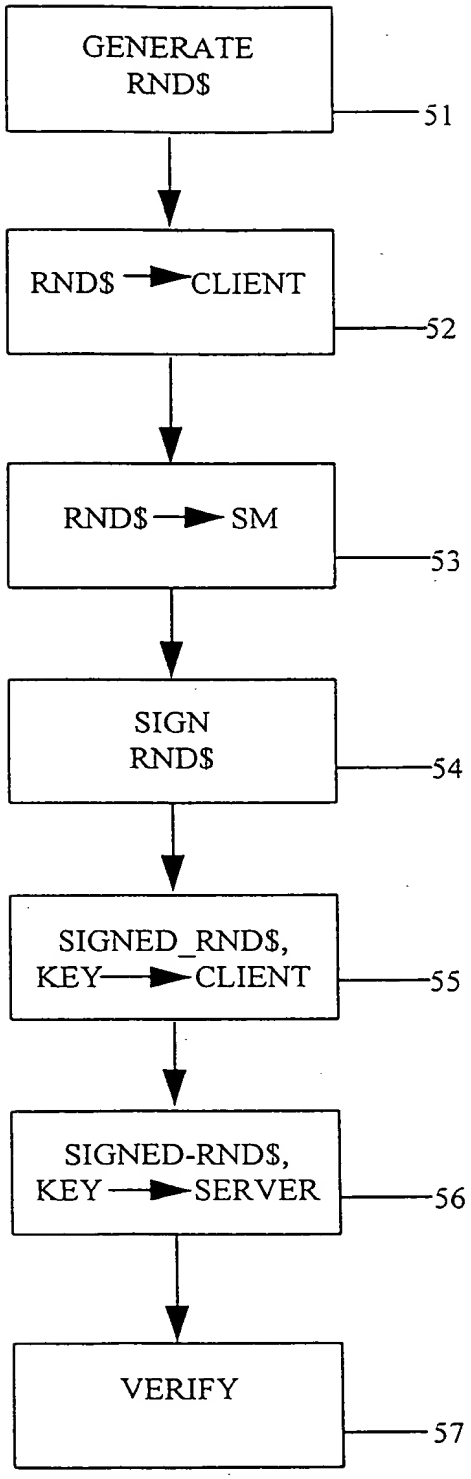

GENERATE RND$
51
RND$ → CLIENT
52
RND$ → SM
53
SIGN RND$
54
SIGNED_RND$, KEY → CLIENT
55
SIGNED-RND$, KEY → SERVER
56
VERIFY
57

Figure 5

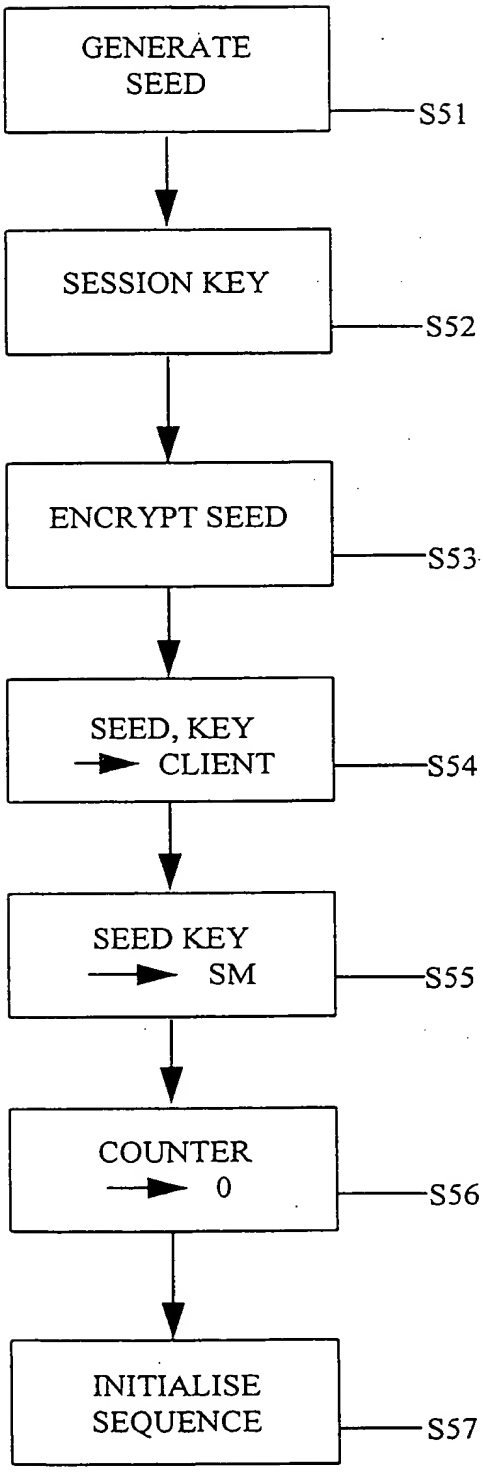

GENERATE SEED
S51
SESSION KEY
S52
ENCRYPT SEED
S53
SEED, KEY → CLIENT
S54
SEED KEY → SM
S55
COUNTER → 0
S56
INITIALISE SEQUENCE
S57

Figure 6

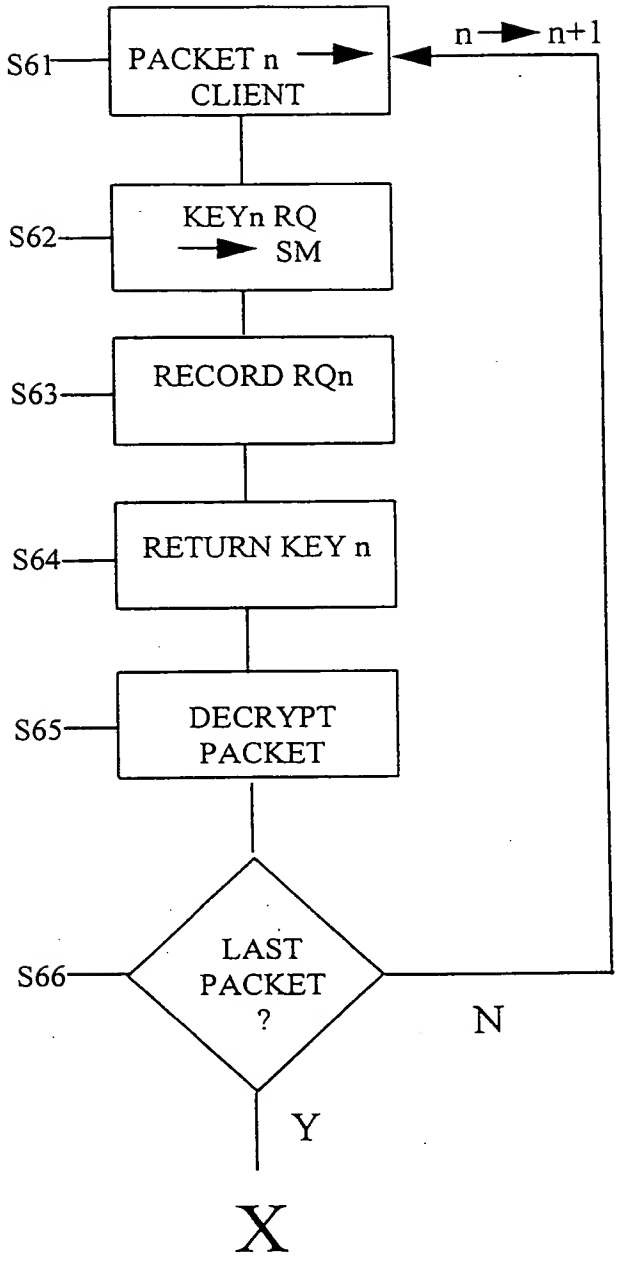
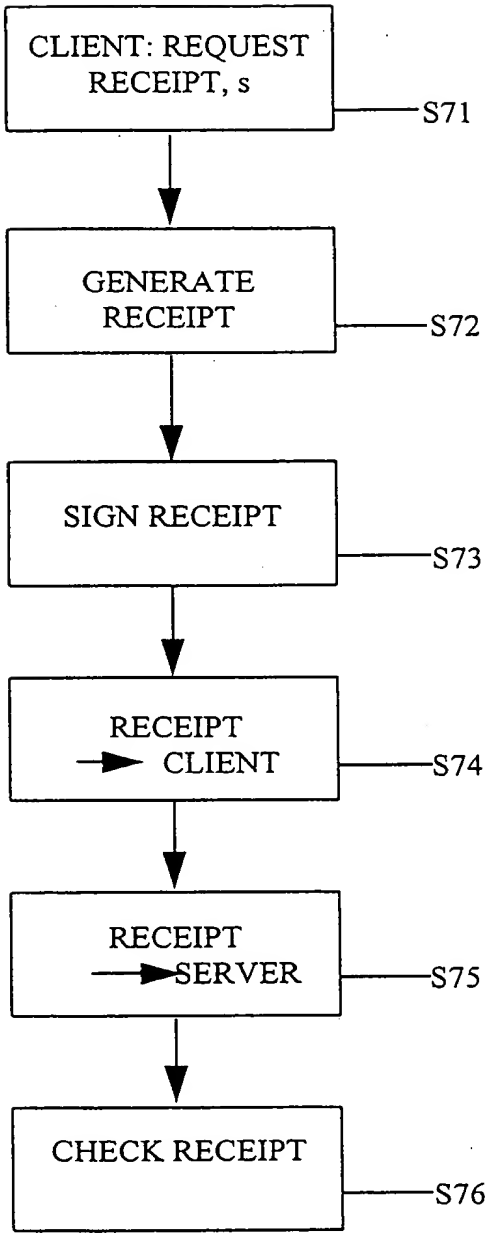

S61
PACKET n
CLIENT
n → n+1
S62
KEYn RQ
SM
S63
RECORD RQn
S64
RETURN KEY n
S65
DECRYPT
PACKET
S66
LAST
PACKET
?
N
Y
X

Figure 7


CLIENT: REQUEST RECEIPT, s
S71
GENERATE RECEIPT
S72
SIGN RECEIPT
S73
RECEIPT → CLIENT
S74
RECEIPT → SERVER
S75
CHECK RECEIPT
S76